Candidat 1 : Jaden — Apprentissage par renforcement & optimisation

1) Résoudre TSP/VRP par RL (pointer networks & policy gradient)

Jeux de données

- **TSPLIB** (TSP & variantes). Instances classiques de référence. comopt.ifi.uni-heidelberg.de+2comopt.ifi.uni-heidelberg.de+2
- **CVRPLIB** (Vehicle Routing). Instances CVRP + meilleurs connus (BKS) pour comparer vos solutions. vrp.galgos.inf.puc-rio.br+1
  (Baselines utiles : HGS-CVRP moderne) GitHub

Algorithmes à tester

- Policy gradient (REINFORCE) avec **pointer network/attention**;
- **Actor-Critic** (A2C/A3C), **PPO**;
- Heuristiques/Meta-heuristiques en baseline (Christofides pour TSP, HGS-CVRP pour CVRP) pour mesurer l'écart à l'optimal/BKS.
  Hypers (exemples) : taille LSTM 128–512, lr 1e-4–3e-3, entropy coef 0–0.02, γ=0.99, batch 64–512, clip-range PPO 0.1–0.3, steps/epoch 5e4–2e5.

Métriques

- **Optimality gap** = (coût_pol − coût_opt)/coût_opt;
- Longueur moyenne de tournée / coût total;
- Temps d'inférence (ms/instance) & stabilité (écart-type sur 10 seeds).

Objectifs de résultats (cibles raisonnables)

- **TSP** ≤ 100 nœuds : gap moyen **< 2–5%** vs optimum TSPLIB;
- **CVRP** taille moyenne : gap **< 5–10%** vs BKS;
- **Inférence** < 50 ms/instance (CPU) sur n ≤ 100.

Livrables

- **Code** (PyTorch) + scripts d'entraînement/évaluation ; intégration **RL4CO** conseillée pour standardiser comparaisons. GitHub+2arXiv+2
- **App Streamlit** : upload d'instance .tsp/.vrp → visualisation de la tournée/route, coût, gap vs baseline, temps d'inférence.
- **Rapport (10–20 p.)** : modélisation MDP, architecture, protocoles d'éval, ablations, analyse erreurs, limites & travail futur.
- **Slides (15–20)** : problème, méthode, démos, résultats clés & leçons.

---

2) CityLearn : RL multi-agents pour lisser la demande énergétique (optimisation coût/peak)

Jeux de données / environnements

- **CityLearn** (env Gymnasium multi-agents + datasets 2022 : charges, météo, prix, intensité carbone). GitHub+2dataverse.tdl.org+2
- **Gymnasium** (API RL maintenue). gymnasium.farama.org+1
- Aperçu & objectifs des challenges CityLearn. citylearn.net+1

Algorithmes à tester

- **MAPPO**, **SAC**, **MADDPG** (multi-agents) vs contrôleurs « rule-based »; **Hypers** : lr 3e-4–1e-3, γ=0.99, τ=0.005, batch 256–1024, target update 1–2×/it, entropy coef 0–0.1.

Métriques

- Score CityLearn : **coût total d'énergie**, **peak-to-average ratio**, **ramping**, **émissions**;
- Stabilité/variance sur 5 runs; robustesse hors-distribution (météo/prix).

Objectifs de résultats

- **-10 à -30%** vs baseline déterministe sur coût total et pic de puissance (selon scénario). *(Bornes inspirées de la littérature CityLearn, à ajuster selon config.)* Proceedings of Machine Learning Research+1

Livrables

- **Code** (agents multi-agents + wrappers d'environnement) ; notebooks d'analyse;
- **Streamlit** : sliders météo/prix → simulate/run → courbes charge/jour, coûts, métriques;
- **Rapport (10–20 p.)** : formulation MDP multi-agents, récompenses, coordination, résultats & ablations;
- **Slides** : story-line du problème réseau → solution RL → gains.